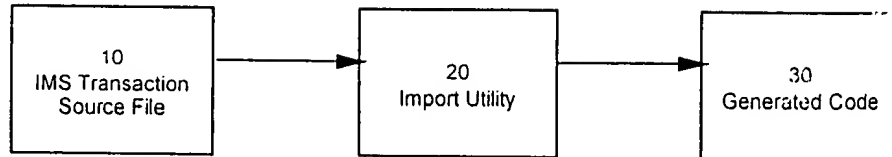


FIGURE 1

Building Application



Running Application

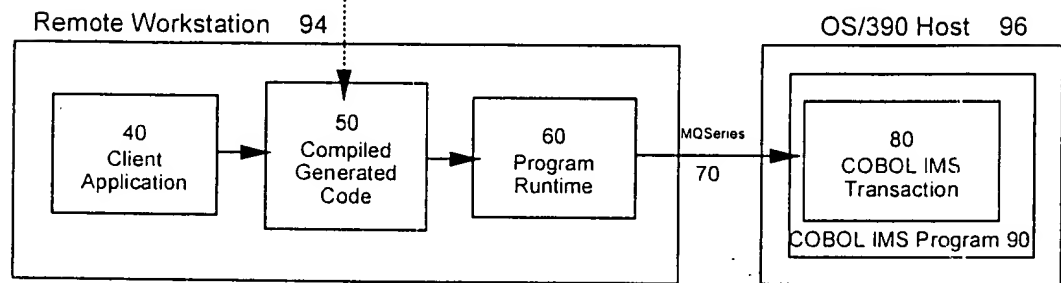
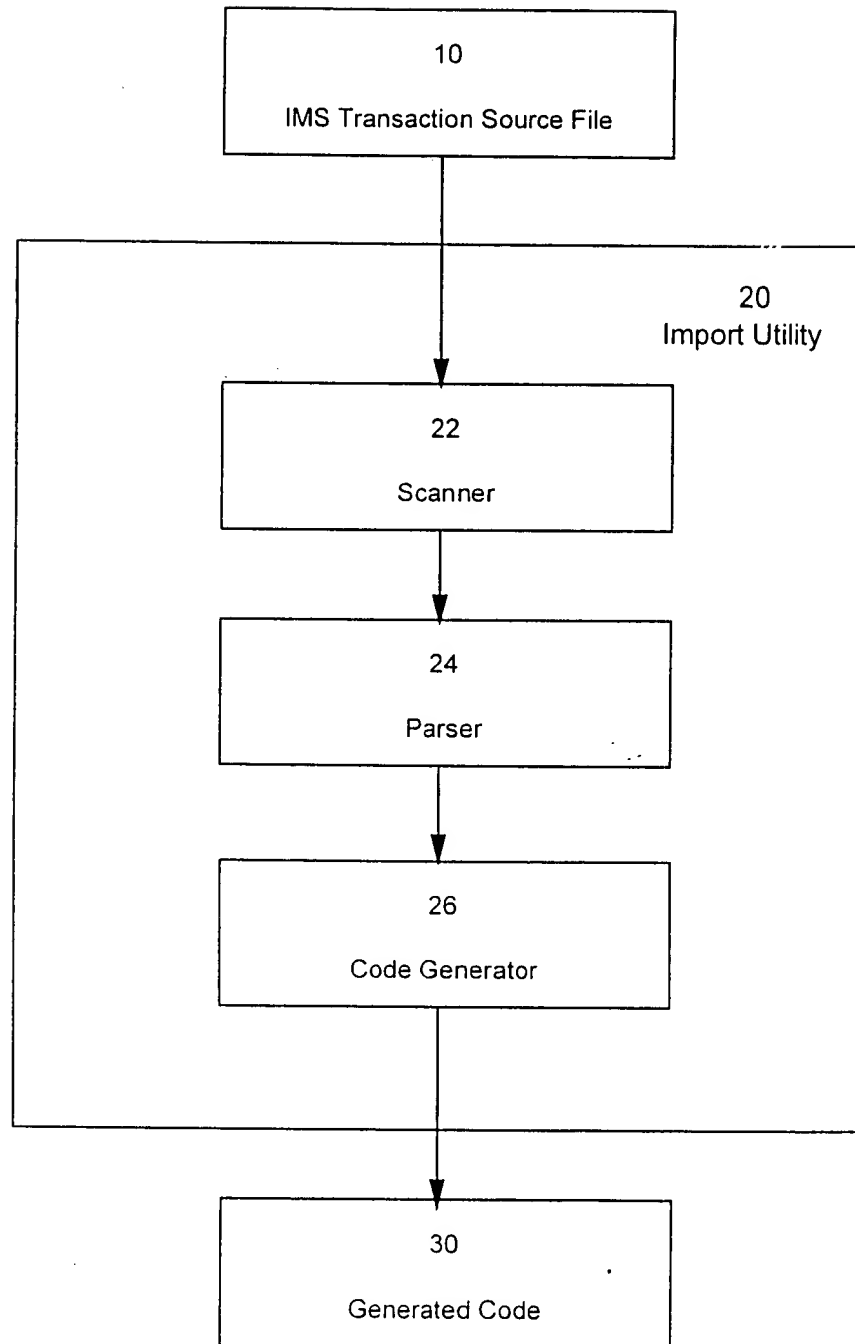


FIGURE 2



mfld5

FIGURE 4A

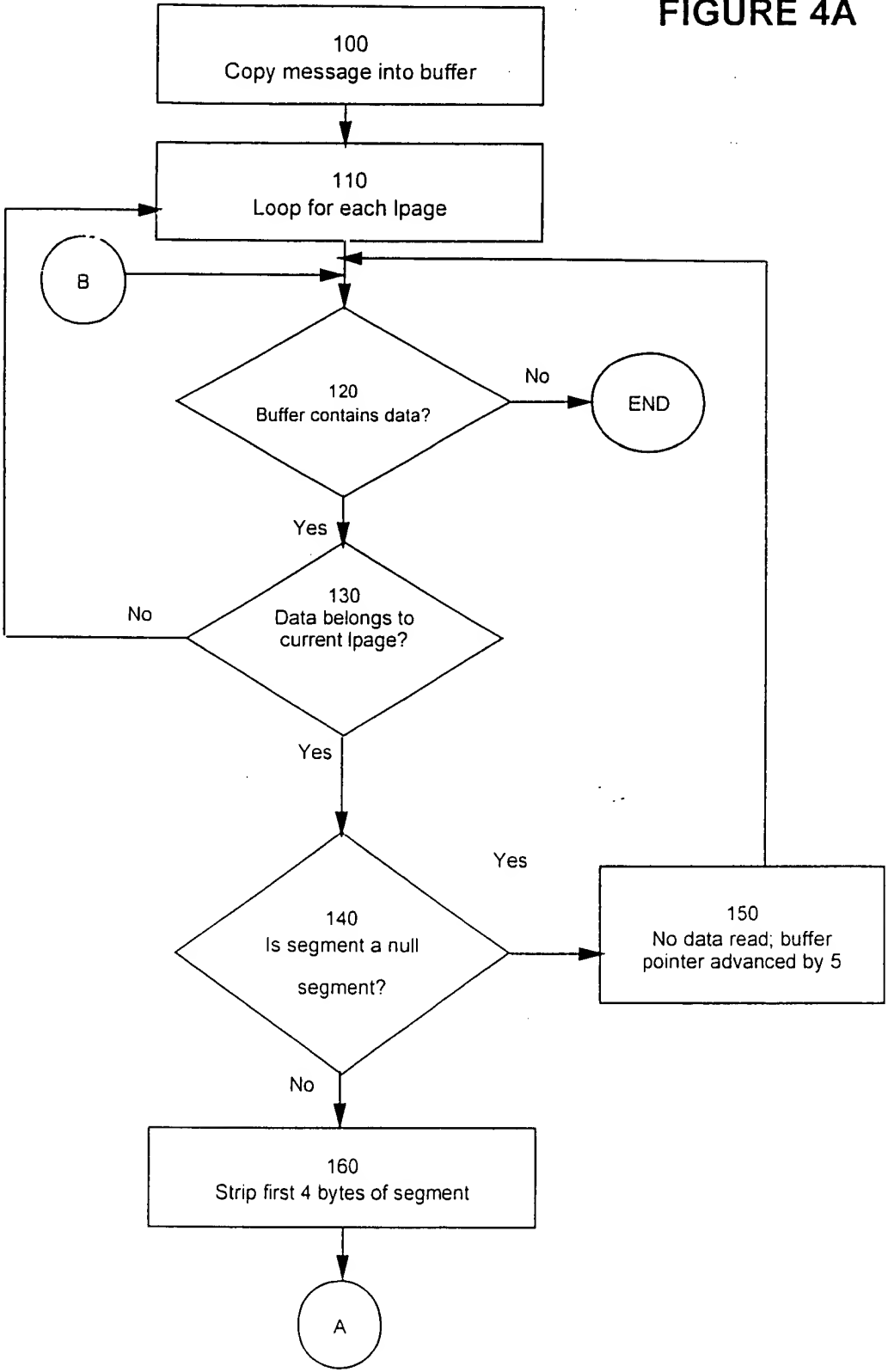


FIGURE 4B

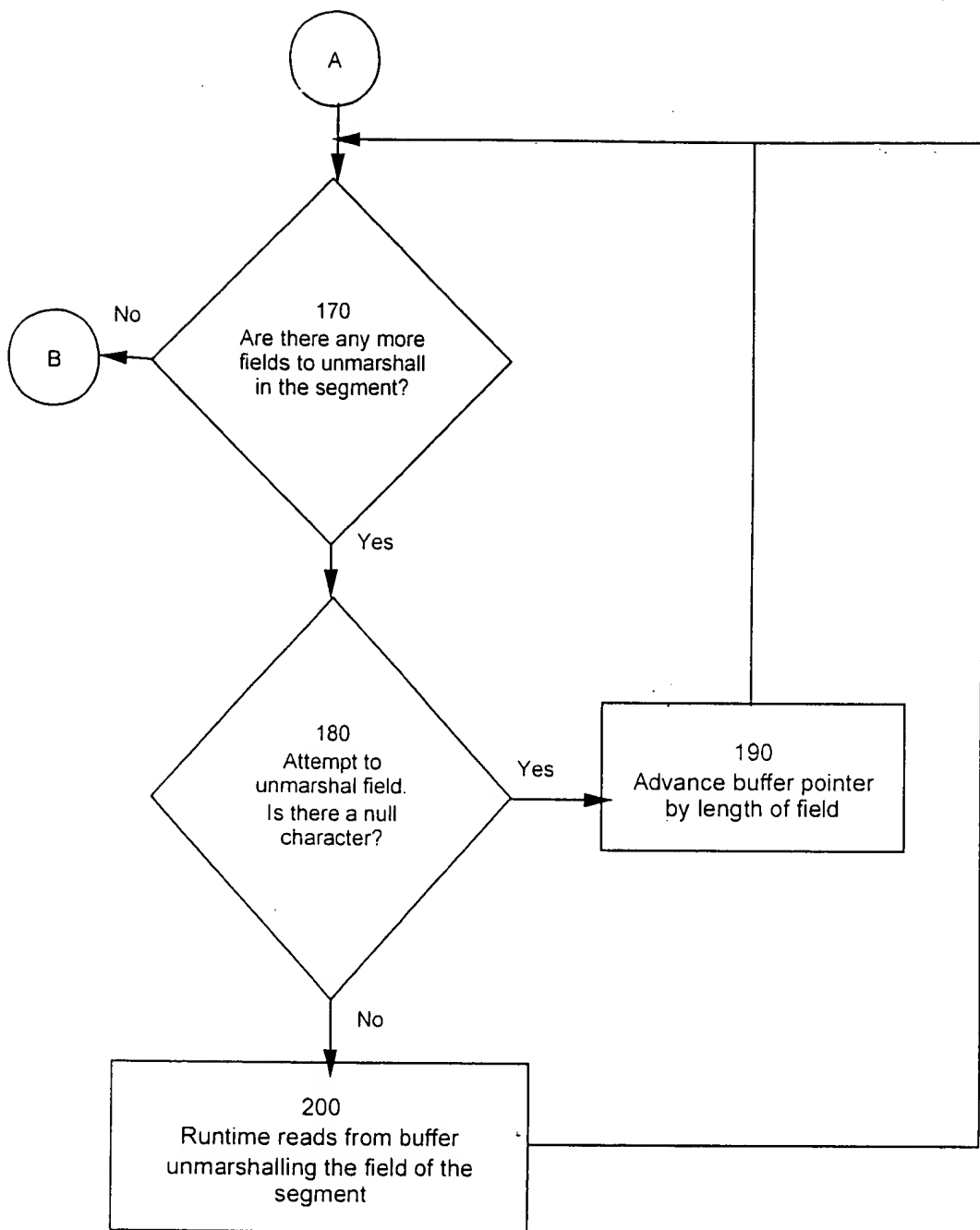


FIGURE 5

IDENTIFICATION DIVISION.  
ENVIRONMENT DIVISION.  
DATA DIVISION.

WORKING-STORAGE SECTION.

77 FILLER PIC X(16) VALUE '\*\*\* BEGIN WS \*\*\*'.

\*\*\*\*\*

\* IMS DECLARATIONS

\*\*\*\*\*

77 DEF-MOD PIC X(8) VALUE 'DFSMO1 ' .  
77 GU-FUNC PIC X(4) VALUE 'GU ' .  
77 GN-FUNC PIC X(4) VALUE 'GN ' .  
77 ISRT-FUNC PIC X(4) VALUE 'ISRT' .  
77 ROLL-FUNC PIC X(4) VALUE 'ROLL' .  
77 DISPLAY-LL PIC 9(5) .  
77 DISPLAY-ZZ PIC 9(5) .

\* begin IMS COBOL input message definition

01 INPUT-MESSAGE.  
02 IN-LL PIC S9(4) COMP.  
02 IN-ZZ PIC S9(4) COMP.  
02 IN-TRANCODE PIC X(8) .  
02 IN-DATA.  
03 OP1 PIC 9(4) DISPLAY.  
03 OP2 PIC 9(4) DISPLAY.  
03 FILLER PIC X(32687) .

\* end IMS COBOL input message definition

\* begin IMS COBOL output message definition

01 OUTPUT-MESSAGE.  
02 OUT-LL PIC S9(4) COMP VALUE +8 .  
02 OUT-ZZ PIC S9(4) COMP VALUE +0 .  
02 RESULT1 PIC 9(4) DISPLAY .

\* end IMS COBOL output message definition

01 DLI-MSG.  
02 DLIMSG-1 PIC X(18) VALUE 'ERROR ON DLI CALL ' .  
02 DLIMSG-CALL PIC X(10) .  
02 DLIMSG-2 PIC X(18) VALUE ' . STATUS CODE WAS ' .  
02 DLIMSG-STAT PIC X(2) .  
02 DLIMSG-3 PIC X(22) VALUE ' ' .

LINKAGE SECTION.

01 IOPCB.  
02 IO-LTERM PIC X(8) .  
02 IO-RESV PIC X(2) .  
  
02 IO-STATUS PIC X(2) .  
02 IO-DATE PIC XXXX .

```

02 IO-TIME          PIC XXXX.
02 IO-SEQNO         PIC XXXX.
02 IO-MODN          PIC X(8).
02 IO-USERID        PIC X(8).
02 IO-GROUPID       PIC X(8).

```

PROCEDURE DIVISION.

```

ENTRY 'DLITCBL' USING IOPCB.
PERFORM GET-IMS-MSG UNTIL (IO-STATUS = 'QC').
GOBACK.

```

```

*****
* GET IMS MESSAGES FROM INPUT QUEUE AND PROCESS
*****
GET-IMS-MSG.

```

```

    MOVE SPACES          TO IN-DATA.
    MOVE SPACES          TO INPUT-MESSAGE.
    CALL 'CBLTDLI' USING GU-FUNC, IOPCB,
                          INPUT-MESSAGE.

```

```

    EVALUATE IO-STATUS
        WHEN ' '
            COMPUTE RESULT1 = OP1 + OP2
            DISPLAY 'OPERAND 1: ' OP1
            DISPLAY 'OPERAND 2: ' OP2
            DISPLAY 'RESULT 1: ' RESULT1
            PERFORM ISRT-MESSAGE
        WHEN 'QC'
            DISPLAY 'QC STATUS CODE RETURNED'
        WHEN NOT 'QC'
            MOVE 'GU#MSGQ' TO DLIMSG-CALL
            MOVE IO-STATUS TO DLIMSG-STAT
            PERFORM D-RETURN-CODE
    END-EVALUATE.

```

```

*****
* ERROR ROUTINE TO CATER FOR UNEXPECTED DLI STATUS CODES
*****

```

```

D-RETURN-CODE.
    DISPLAY DLI-MSG.
    MOVE 0 TO RESULT1.
    PERFORM ISRT-MESSAGE.
    GOBACK.

```

```

*****
* ROUTINE TO INSERT A MESSAGE TO THE TERMINAL
*****

```

ISRT-MESSAGE.

```
      IF IO-MODN = 'MQIMSVS'
*      MOVE 'MQIMSVS'          TO DEF-MOD
      MOVE 'T004002'          TO DEF-MOD
      ELSE
*      MOVE 'DFSM01'           TO DEF-MOD.
      MOVE 'T004002'           TO DEF-MOD.
      MOVE 'T004002'           TO IO-MODN
      DISPLAY 'DEF-MOD: ' DEF-MOD.
      DISPLAY 'IO-MODN: ' IO-MODN.
      CALL 'CBLTDLI' USING ISRT-FUNC, IOPCB,
      OUTPUT-MESSAGE, DEF-MOD.
      IF IO-STATUS NOT = SPACES
      MOVE 'ISRT#M' TO DLIMSG-CALL
      MOVE IO-STATUS TO DLIMSG-STAT
      DISPLAY DLI-MSG
      CALL 'CBLTDLI' USING ROLL-FUNC, IOPCB.
```



FIGURE 6

```
MYCLASS.HPP
#ifndef _MYCLASS_HPP_
#define _MYCLASS_HPP_
//
// FILE NAME: myclass.hpp
//

#include <idaif.hpp>
#include <idaifs.hpp>
#include <idabndg.hpp>
#include <idach.hpp>
#include <idamw.hpp>
#include <idauuid.hpp>
#include <idaexc.hpp>

#include <ibag.h>
#include <idaifb.hpp>
#include <ixdims.hpp>
#include <ixdimstr.hpp>
#include <ixdimsac.hpp>
#include <ixdimsca.hpp>
#include <ixdimsco.hpp>
#include <ixdimsns.hpp>
#include <ixdimspo.hpp>
#include <ixdimspf.hpp>
#include "myclass.imc"

class myclass :
    public IStandardNotifier,
    public IXDMQIMSConversation,
    public IDAInterfaceBase,
    public virtual myclassDefinition
{
public:

    myclass()
    {
        // Tell object to use IMS stubs
        try {
            this->getInterfaceStub (IXDMQIMS::instance());
        }
        catch (IDANoInterfaceStubException& ex) {
            this->addInterfaceStub (*(new ims_myclass_cstub),
IXDMQIMS::instance());
        }

        // Set up MQI persistence attributes
    }
};
```

```

this->addAttributes (&IXDMQIMSPersistenceOff::instance());
this->addAttributes (addFv_name(), &IXDMQIMSPersistenceOff::instance());

pimsco_ = new IXDMQIMSCnvAttr((IXDMQIMSConversation *) this);
this->addAttributes( pimsco_ );

pimsac_ = new IXDMQIMSAccess((char *)getName());
this->addAttributes(pimsac_);
IXDMQIMSTransaction *trans;

trans = new IXDMQIMSTransaction("TCLI0040",
                                '1',
                                (char *)getName(),
                                'C'

                                ,0x20
                                ,""
                                ,1
                                ,0
                                ,0x20
                                ,""
                                ,1
                                ,0

);

imstats_.add( (void *) trans );
this->addAttributes (addFv_name(), trans);

imsns_ = new IXDMQIMSNameService(&IXDMQIMS::instance(), getPrimaryId(),
                                getSecondaryId(), (char *)getName());

IXDMQIMS::instance().addNameService (imsns_);

// Import from correct place
this->importBindings(IXDMQIMS::instance());
}
~myclass() {
    if(NULL != pimsco_)
        delete pimsco_;
    if(NULL != pimsac_)
        delete pimsac_;

    IXDMQIMSTransaction *tr;
    while(!imstats_.isEmpty()) {
        tr = (IXDMQIMSTransaction *) imstats_.anyElement();
        imstats_.remove(tr);
        delete tr;
    }

    IXDMQIMS::instance().removeNameService (imsns_);
    delete imsns_;
}

```

```

virtual void add ( myclass_add_I* inputMsg, myclass_add_O* outputMsg )
{
    IDABinding* binding;

    if (blist.numberOfElements() == 0)
        ((myclass *)this)->blist =
            ((myclass *)this)->importBindings();

    IBag<IDABinding *>::Cursor current(blist);
    current.setToFirst();
    if (!current.isValid()) {
        throw IDANoBindingException(*((myclass *)this));
    }
    binding = blist.elementAt(current);

    IDACallHandle ch = binding->mw()->createCallHandle();
    ch.set(((myclass *)this)->getPrimaryId(),
        ((myclass *)this)->getSecondaryId(),
        addFv_name(), binding);

    ((myclass *)this)->setAttributes(ch);
    ch.genRequest();
    try {
        ((ims_myclass_cstub *)((myclass
*)this)->getInterfaceStub(*binding->mw()))->add(ch, inputMsg, outputMsg);
        ch.genConfirm();
    } catch (IException& ex) {
        ch.genConfirm();
        throw (ex);
    }
}

private:

    IXDMQIMSNameService *imsns_;
    IXDMQIMSConvAttr *pimsco_;
    IXDMQIMSAccess *pimsac_;
    IBag<void *> imstats_;
};

#endif

MYCLASS.IMC
#ifdef _MYCLASS_IMC_

//
// FILE NAME: myclass.imc

```

```

//

#include <ixdimsex.hpp>
#include <idacicch.hpp>
#include <ixdimesch.hpp>

#include "myclass.hpd"

class ims_myclass_cstub :
    public IDAInterfaceStub,
    public myclass_stub {
public:

    static ims_myclass_cstub& instance()
    {
        if (instance_ == NULL)
            instance_ = new ims_myclass_cstub;
        return (*instance_);
    }

    void add (IDACallHandle& ch, myclass_add_I* inputMsg, myclass_add_O*
outputMsg)
    {
        ch.start();
        ch << (myclass_add_I&) (*inputMsg);
        ch.transceive();
        ch >> (myclass_add_O&) (*outputMsg);
        outputMsg->notify();
        ch.done();
    }

private :

    static ims_myclass_cstub* instance_;
};

ims_myclass_cstub* ims_myclass_cstub::instance_ = NULL;

#endif

MYCLASS.HPD
#ifndef _MYCLASS_HPD_
#define _MYCLASS_HPD_

// Class: myclass
//
// FILE NAME: myclass.hpd
//

```

```

#include <istring.hpp>
#include <idauid.hpp>
#include <idaifd.hpp>
#include <idach.hpp>

#include "myclass.imd"

class myclassDefinition : public virtual IDAInterfaceDefinition
{
public:

    myclassDefinition()
    {
        setSecondaryId (IDAUid::nil.toString());
        setName ("myclass");
    }

    IDAUid getPrimaryId() {
        return IDAUid("165c9ec5-2a1d-02f0-8000-400011528584");
    }

    IString addFv_name() const {
        return IString("void myclass::add( myclass_add_I* inputMsg,
myclass_add_O* outputMsg )");
    }

private:

};
class myclass_stub : public virtual myclassDefinition
{
public:
    virtual void add (IDACallHandle& ch, myclass_add_I* inputMsg,
myclass_add_O* outputMsg) = 0;
};

#endif

MYCLASS.IMD
#ifndef _MYCLASS_IMD_
#define _MYCLASS_IMD_
//
// FILE NAME: myclass.imd
//

#include <istdntfy.hpp>
#include <istring.hpp>
#include <idach.hpp>
#include <idacicch.hpp>

```

```

#include <inotifev.hpp>
#include <istdntfy.hpp>
#include <istring.hpp>
#include <ivseq.h>
#include <ixdimsex.hpp>
#include <ixdimstg.hpp>

```

```

class myclass_add_Lpage1_args {
public:

```

```

    myclass_add_Lpage1_args() {
        op1_ = 0;
        op2_ = 0;
        op1_flag_ = 0;
        op2_flag_ = 0;
    }

```

```

    unsigned short int op1_;
    IBoolean op1_flag_;
    unsigned short int op2_;
    IBoolean op2_flag_;
};

```

```

inline IDACallHandle& operator<< (IDACallHandle& ch, const
myclass_add_Lpage1_args& d)
{
    ch.offsetLL();
    ch << IDACICSCallHandleInternal::PIC(ch, "9(4)");
    ch << IDACICSCallHandleInternal::AUG(ch, "DISPLAY_NUMERIC");
    ch << (unsigned short int &)d.op1_;
    ch << IDACICSCallHandleInternal::PIC(ch, "9(4)");
    ch << IDACICSCallHandleInternal::AUG(ch, "DISPLAY_NUMERIC");
    ch << (unsigned short int &)d.op2_;
    ch.setLL();
    return ch;
}

```

```

class myclass_add_Lpage1 : public IStandardNotifier {
public:

```

```

    myclass_add_Lpage1()
    {
    }

```

```

    ~myclass_add_Lpage1()
    {
    }

```

```

    myclass_add_Lpage1& operator= (const myclass_add_Lpage1& aLpage1_)
    {
        setop1(aLpage1_.op1());
    }

```

```

        setop2(aLpage1_.op2());
        return *this;
    }

    myclass_add_Lpage1(const myclass_add_Lpage1& aLpage1_)
    {
        setop1(aLpage1_.op1());
        setop2(aLpage1_.op2());
    }

    void notify()
    {
        if (args_.op1_flag_) notifyObservers(INotificationEvent(setop1Id,
*this));
        if (args_.op2_flag_) notifyObservers(INotificationEvent(setop2Id,
*this));
    }

    unsigned short int op1 () const
    {
        return (args_.op1_);
    }

    static INotificationId setop1Id;
    myclass_add_Lpage1& setop1 (const unsigned short int& aop1)
    {
        args_.op1_ = aop1;
        notifyObservers (INotificationEvent(setop1Id, *this));
        return *this;
    }

    unsigned short int op2 () const
    {
        return (args_.op2_);
    }

    static INotificationId setop2Id;
    myclass_add_Lpage1& setop2 (const unsigned short int& aop2)
    {
        args_.op2_ = aop2;
        notifyObservers (INotificationEvent(setop2Id, *this));
        return *this;
    }

    myclass_add_Lpage1_args args_;

};

inline IDACallHandle& operator<< (IDACallHandle& ch, const myclass_add_Lpage1&
d)
{

```

```

ch << (myclass_add_Lpagel_args &)d.args_;

return ch;
}

class myclass_add_I : public IStandardNotifier {
public:

    myclass_add_I()
    {
        Lpagel_.addAsFirst(new myclass_add_Lpagel);
        Lpagel_flag_ = 0;
    }
    ~myclass_add_I()
    {
        while(!Lpagel_.isEmpty()) {
            myclass_add_Lpagel* anElement = Lpagel_.firstElement();
            Lpagel_.removeFirst();
            delete anElement;
        }
    }
    myclass_add_I& operator= (const myclass_add_I& ainputMsg)
    {
        while(!Lpagel_.isEmpty()) {
            myclass_add_Lpagel* anElement = Lpagel_.firstElement();
            Lpagel_.removeFirst();
            delete anElement;
        }
        IVSequence<myclass_add_Lpagel*>::Cursor cursor(ainputMsg.Lpagel_);
        forCursor(cursor)
            Lpagel_.addAsLast(new myclass_add_Lpagel(*cursor.element()));
        notifyObservers (INotificationEvent(Lpagel_Id, *this));
        Lpagel_flag_ = ainputMsg.Lpagel_flag_;
        return *this;
    }

    myclass_add_I(const myclass_add_I& ainputMsg)
    {
        IVSequence<myclass_add_Lpagel*>::Cursor cursor(ainputMsg.Lpagel_);
        forCursor(cursor)
            Lpagel_.addAsLast(new myclass_add_Lpagel(*cursor.element()));
        notifyObservers (INotificationEvent(Lpagel_Id, *this));
        Lpagel_flag_ = ainputMsg.Lpagel_flag_;
    }

    void notify()
    {
        if (Lpagel_flag_) {
            notifyObservers (INotificationEvent(Lpagel_seqId, *this));
            notifyObservers (INotificationEvent(Lpagel_Id, *this));
            notifyObservers (INotificationEvent(setopId, *this));
        }
    }
}

```



```

        notifyObservers (INotificationEvent(setop2Id, *this));
    }
}

IVSequence<myclass_add_Lpage1*> Lpage1_seq() {
    return (Lpage1_);
}

static INotificationId Lpage1_seqId;
myclass_add_I & setLpage1_seq( IVSequence<myclass_add_Lpage1*> *
aLpage1_) {
    while(!Lpage1_.isEmpty()) {
        myclass_add_Lpage1* anElement = Lpage1_.firstElement();
        Lpage1_.removeFirst();
        delete anElement;
    }
    IVSequence<myclass_add_Lpage1*>::Cursor cursor(*aLpage1_);
    forCursor(cursor)
        Lpage1_.addAsLast(new myclass_add_Lpage1(*cursor.element()));
    notifyObservers (INotificationEvent(Lpage1_seqId, *this));
    return *this;
}

myclass_add_Lpage1 Lpage1__() {
    return (*(Lpage1_.firstElement()));
}

static INotificationId Lpage1_Id;
myclass_add_I & setLpage1__( myclass_add_Lpage1* aLpage1_) {
    while(!Lpage1_.isEmpty()) {
        myclass_add_Lpage1* anElement = Lpage1_.firstElement();
        Lpage1_.removeFirst();
        delete anElement;
    }
    Lpage1_.add(new myclass_add_Lpage1(*aLpage1_));
    notifyObservers (INotificationEvent(Lpage1_Id, *this));
    return *this;
}

    unsigned short int      opl (      )      const
{
    return ( Lpage1_.firstElement()->opl());
}

    static INotificationId setoplId;
myclass_add_I &      setopl (      const unsigned short int& aopl
)
{
    Lpage1_.firstElement()->setopl(aopl);
    notifyObservers (INotificationEvent(setoplId, *this));
    return *this;
}

    unsigned short int      op2 (      )      const
{

```

```

        return ( Lpage1_.firstElement()->op2());
    }

    static INotificationId setop2Id;
    myclass_add_I &      setop2 (      const unsigned short int& aop2
)
    {
        Lpage1_.firstElement()->setop2(aop2);
        notifyObservers (INotificationEvent(setop2Id, *this));
        return *this;
    }

    IVSequence<myclass_add_Lpage1*> Lpage1_;
    IBoolean Lpage1_flag_;

};

inline IDACallHandle& operator<< (IDACallHandle& ch, myclass_add_I& d)
{
    IVSequence<myclass_add_Lpage1 *>::Cursor cursor(d.Lpage1_);
    forCursor(cursor) {
        ch << d.Lpage1_.elementAt(cursor);
        d.Lpage1_flag_ = 1;
    }

    ch.setPageBit();

    ch.stripNullSegments();

    return ch;
}

class myclass_add_Lpage2_result {
public:

    myclass_add_Lpage2_result() {
        result1_ = 0;
        result1_flag_ = 0;
    }

    unsigned short int result1_;
    IBoolean result1_flag_;
};

inline IDACallHandle& operator>> (IDACallHandle& ch,
myclass_add_Lpage2_result& d)
{
    if (ch.notNullSegment() && ch.stripLL()) {
        ch >> IDACICSCallHandleInternal::PIC(ch, "9(4)");
        ch >> IDACICSCallHandleInternal::AUG(ch, "DISPLAY_NUMERIC");
        ch >> (unsigned short int &)d.result1_;
        if (ch.fieldIsSet()) d.result1_flag_ = 1;
    }
}

```

```

    }

    return ch;
}

class myclass_add_Lpage2 : public IStandardNotifier {
public:

    myclass_add_Lpage2()
    {
    }

    ~myclass_add_Lpage2()
    {
    }

    myclass_add_Lpage2& operator= (const myclass_add_Lpage2& aLpage2_)
    {
        setresult1(aLpage2_.result1());
        return *this;
    }

    myclass_add_Lpage2(const myclass_add_Lpage2& aLpage2_)
    {
        setresult1(aLpage2_.result1());
    }

    void notify()
    {
        if (result_.result1_flag_)
            notifyObservers(INotificationEvent(setresult1Id, *this));
    }

    unsigned short int result1 () const
    {
        return (result_.result1_);
    }

    static INotificationId setresult1Id;
    myclass_add_Lpage2& setresult1 (const unsigned short int& aresult1)
    {
        result_.result1_ = aresult1;
        notifyObservers (INotificationEvent(setresult1Id, *this));
        return *this;
    }

    myclass_add_Lpage2_result result_;
};

```

```

inline IDACallHandle& operator>> (IDACallHandle& ch, myclass_add_Lpage2& d)
{
    ch >> (myclass_add_Lpage2_result &)d.result_;

    return ch;
}

class myclass_add_O : public IStandardNotifier {
public:

    myclass_add_O()
    {
        Lpage2_.addAsFirst(new myclass_add_Lpage2);
        Lpage2_flag_ = 0;
    }
    ~myclass_add_O()
    {
        while(!Lpage2_.isEmpty()) {
            myclass_add_Lpage2* anElement = Lpage2_.firstElement();
            Lpage2_.removeFirst();
            delete anElement;
        }
    }
    myclass_add_O& operator= (const myclass_add_O& aoutputMsg)
    {
        while(!Lpage2_.isEmpty()) {
            myclass_add_Lpage2* anElement = Lpage2_.firstElement();
            Lpage2_.removeFirst();
            delete anElement;
        }
        IVSequence<myclass_add_Lpage2*>::Cursor cursor(aoutputMsg.Lpage2_);
        forCursor(cursor)
            Lpage2_.addAsLast(new myclass_add_Lpage2(*cursor.element()));
        notifyObservers (INotificationEvent(Lpage2_Id, *this));
        Lpage2_flag_ = aoutputMsg.Lpage2_flag_;
        return *this;
    }
    myclass_add_O(const myclass_add_O& aoutputMsg)
    {
        IVSequence<myclass_add_Lpage2*>::Cursor cursor(aoutputMsg.Lpage2_);
        forCursor(cursor)
            Lpage2_.addAsLast(new myclass_add_Lpage2(*cursor.element()));
        notifyObservers (INotificationEvent(Lpage2_Id, *this));
        Lpage2_flag_ = aoutputMsg.Lpage2_flag_;
    }
    void notify()
    {
        if (Lpage2_flag_) {
            notifyObservers (INotificationEvent(Lpage2_seqId, *this));
            notifyObservers (INotificationEvent(Lpage2_Id, *this));
            notifyObservers (INotificationEvent(setresultId, *this));
        }
    }

```

```

    }
}

IVSequence<myclass_add_Lpage2*> Lpage2_seq() {
    return (Lpage2_);
}

static INotificationId Lpage2_seqId;
myclass_add_O & setLpage2_seq( IVSequence<myclass_add_Lpage2*> *
aLpage2_ ) {
    while(!Lpage2_.isEmpty()) {
        myclass_add_Lpage2* anElement = Lpage2_.firstElement();
        Lpage2_.removeFirst();
        delete anElement;
    }
    IVSequence<myclass_add_Lpage2*>::Cursor cursor(*aLpage2_);
    forCursor(cursor)
        Lpage2_.addAsLast(new myclass_add_Lpage2(*cursor.element()));
    notifyObservers (INotificationEvent(Lpage2_seqId, *this));
    return *this;
}

myclass_add_Lpage2 Lpage2__() {
    return (*(Lpage2_.firstElement()));
}

static INotificationId Lpage2_Id;
myclass_add_O & setLpage2__( myclass_add_Lpage2* aLpage2_ ) {
    while(!Lpage2_.isEmpty()) {
        myclass_add_Lpage2* anElement = Lpage2_.firstElement();
        Lpage2_.removeFirst();
        delete anElement;
    }
    Lpage2_.add(new myclass_add_Lpage2(*aLpage2_));
    notifyObservers (INotificationEvent(Lpage2_Id, *this));
    return *this;
}

    unsigned short int      result1 (      )      cons:
{
    return ( Lpage2_.firstElement()->result1());
}

    static INotificationId setResult1Id;
myclass_add_O &      setResult1 (      const unsigned short int&
aresult1
    )
{
    Lpage2_.firstElement()->setResult1(aresult1);
    notifyObservers (INotificationEvent(setResult1Id, *this));
    return *this;
}

IVSequence<myclass_add_Lpage2*> Lpage2_;
IBoolean Lpage2_flag_;
};

inline IDACallHandle& operator>> (IDACallHandle& ch, myclass_add_O& d)
{

```

```

while (ch.notAtEndOfBuffer()) {
    while(!d.Lpage2_.isEmpty()) {
        myclass_add_Lpage2* anElement = d.Lpage2_.firstElement();
        d.Lpage2_.removeFirst();
        delete anElement;
    }
    myclass_add_Lpage2 tempLpage2_;
    while (ch.notAtEndOfBufferOrSeq(0, 1, "0")) {
        ch >> (myclass_add_Lpage2 &) tempLpage2_;
        d.Lpage2_.addAsLast(new myclass_add_Lpage2(tempLpage2_));
        d.Lpage2_flag_ = 1;
    }
    if (!ch.unmarshallLPAGE()) {
        if (ch.notNullSegment()) {
            throw IXDMQIMSEException((const char *)
                IMessageText(160, IXDMQIMS_MSG_FILE) );
        } else {
            throw IXDMQIMSEException((const char *)
                IMessageText(161, IXDMQIMS_MSG_FILE) );
        }
    }
}
return ch;
}
#endif

MYCLASS.VBE
//VBBeginPartInfo: myclass
//VBParent: IStandardNotifier
//VBIncludes: "myclass.hpp" _MYCLASS_HPP_
//VBPartDataFile: myclass.vbb
//VBConstructor: myclass()
//VBComposerInfo: nonvisual
//VBLibFile: idacom.lib
//VB: idaims10.lib
//VBEvent: ready,"ready", readyId
//VBAction: add,"add method",void,add(myclass_add_I* inputMsg,myclass_add_O*
outputMsg)
//VBPreferredFeatures: add, enabledForNotification, this
//VBEndPartInfo: myclass
//VBBeginPartInfo: myclass_add_I
//VBParent: IStandardNotifier
//VBIncludes: "myclass.hpp" _MYCLASS_HPP_
//VBPartDataFile: myclass.vbb
//VBConstructor: myclass_add_I()
//VBComposerInfo: nonvisual
//VBEvent: ready,"ready", readyId
//VBAttribute: Lpage1_seq, "Lpage1_seq", IVSequence<myclass_add_Lpage1*>,
IVSequence<myclass_add_Lpage1*> Lpage1_seq(), myclass_add_I & setLpage1_seq(
IVSequence<myclass_add_Lpage1*> * aLpage1_), Lpage1_seqId

```

```

//VBAttribute: Lpage1_, "Lpage1_", myclass_add_Lpage1, myclass_add_Lpage1
Lpage1_(), myclass_add_I & setLpage1_( myclass_add_Lpage1* aLpage1_),
Lpage1_Id
//VBAttribute: op1_, "op1", unsigned short int, unsigned short int
op1(), myclass_add_I & setop1(unsigned short int aop1), setop1Id
//VBAttribute: op2_, "op2", unsigned short int, unsigned short int
op2(), myclass_add_I & setop2(unsigned short int aop2), setop2Id
//VBPreferredFeatures: Lpage1_seq, Lpage1_, op1_, op2_, this
//VBEndPartInfo: myclass_add_I
//VBBeginPartInfo: myclass_add_Lpage1
//VBParent: IStandardNotifier
//VBIncludes: "myclass.hpp" _MYCLASS_HPP_
//VBPartDataFile: myclass.vbb
//VBConstructor: myclass_add_Lpage1()
//VBComposerInfo: nonvisual
//VBEvent: ready, "ready", readyId
//VBAction: operator=, "Assigns
myclass_add_Lpage1", myclass_add_Lpage1&, operator=(const myclass_add_Lpage1&
aLpage1_)
//VBAttribute: op1_, "op1", unsigned short int, unsigned short int op1(),
myclass_add_Lpage1 & setop1(unsigned short int aop1), setop1Id
//VBAttribute: op2_, "op2", unsigned short int, unsigned short int op2(),
myclass_add_Lpage1 & setop2(unsigned short int aop2), setop2Id
//VBPreferredFeatures: operator=, op1_, op2_, this
//VBEndPartInfo: myclass_add_Lpage1
//VBBeginPartInfo: myclass_add_O
//VBParent: IStandardNotifier
//VBIncludes: "myclass.hpp" _MYCLASS_HPP_
//VBPartDataFile: myclass.vbb
//VBConstructor: myclass_add_O()
//VBComposerInfo: nonvisual
//VBEvent: ready, "ready", readyId
//VBAttribute: Lpage2_seq, "Lpage2_seq", IVSequence<myclass_add_Lpage2*>,
IVSequence<myclass_add_Lpage2*> Lpage2_seq(), myclass_add_O & setLpage2_seq(
IVSequence<myclass_add_Lpage2*> * aLpage2_), Lpage2_seqId
//VBAttribute: Lpage2_, "Lpage2_", myclass_add_Lpage2, myclass_add_Lpage2
Lpage2_(), myclass_add_O & setLpage2_( myclass_add_Lpage2* aLpage2_),
Lpage2_Id
//VBAttribute: result1_, "result1", unsigned short int, unsigned short int
result1(), myclass_add_O & setresult1(unsigned short int aresult1), setresult1Id
//VBPreferredFeatures: Lpage2_seq, Lpage2_, result1_, this
//VBEndPartInfo: myclass_add_O
//VBBeginPartInfo: myclass_add_Lpage2
//VBParent: IStandardNotifier
//VBIncludes: "myclass.hpp" _MYCLASS_HPP_
//VBPartDataFile: myclass.vbb
//VBConstructor: myclass_add_Lpage2()
//VBComposerInfo: nonvisual
//VBEvent: ready, "ready", readyId

```

```

//VBAction: operator=,"Assigns
myclass_add_Lpage2",myclass_add_Lpage2&,operator=(const myclass_add_Lpage2&
aLpage2_)
//VBAttribute: result1_,"result1",unsigned short int,unsigned short int
result1(), myclass_add_Lpage2 & setresult1(unsigned short int
aresult1),setresult1Id
//VBPreferredFeatures: operator=, result1_, this
//VBEndPartInfo: myclass_add_Lpage2

```

MYCLASS.CPP

```

//
// FILE NAME: myclass.cpp
//

```

```

#include "myclass.imd"

```

```

INotificationId myclass_add_I::Lpage1_seqId = "Lpage1_seqId";
INotificationId myclass_add_I::Lpage1_Id = "Lpage1_Id";
INotificationId myclass_add_I::setop1Id = "setop1Id";
INotificationId myclass_add_Lpage1::setop1Id = "setop1Id";
INotificationId myclass_add_I::setop2Id = "setop2Id";
INotificationId myclass_add_Lpage1::setop2Id = "setop2Id";
INotificationId myclass_add_O::Lpage2_seqId = "Lpage2_seqId";
INotificationId myclass_add_O::Lpage2_Id = "Lpage2_Id";
INotificationId myclass_add_O::setresult1Id = "setresult1Id";
INotificationId myclass_add_Lpage2::setresult1Id = "setresult1Id";

```